PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-163310
(43)Date of publication of application : 07.06.2002

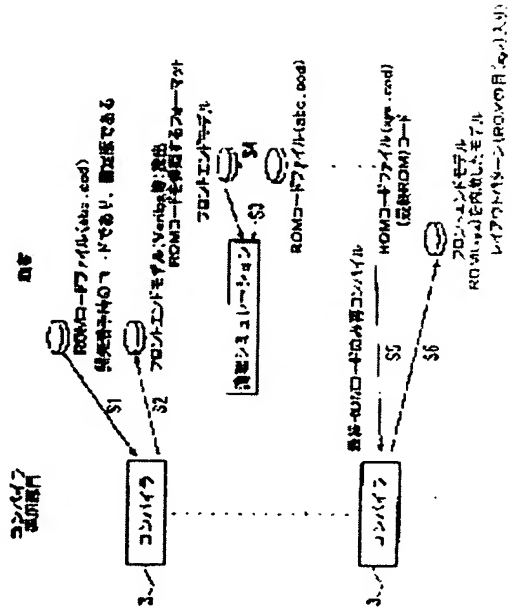
(51)Int.Cl. G06F 17/50

(21)Application number : 2000-361456
(22)Date of filing : 28.11.2000

(71)Applicant : HITACHI LTD
(72)Inventor : AIZAWA SHINYA
SHIMADA SHIGERU

(54) DESIGNING METHOD FOR SEMICONDUCTOR INTEGRATED CIRCUIT

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a designing method for semiconductor integrated circuit, which can easily transit to a corrected logic simulation and can easily manage the version of stored data as well when the stored data in a ROM module are corrected in logic design.
SOLUTION: When generating a data file expressing the ROM module up to a logic level while using a hardware description language, the information of bit data stored in the memory cell of the ROM module is stored in a file different from the data file of the module and in the form referring to the subordinate data file recorded with these bit data, the data file of the ROM module is generated. Further, the layout design of the mask patterns of the ROM module is executed so that the file name of the data file with the stored data recorded thereon can be visibly engraved on the inter-layer oxide film provided with a contact hole for determining the data value of the ROM module.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2002－163310
（P2002－163310A）
(43)公開日　平成14年６月７日(2002.6.7)

(51)Int.Cl.'	識別記号	ＦＩ	テーマコード'(参考)
Ｇ０６Ｆ　17/50	６５２	Ｇ０６Ｆ　17/50	６５２Ｃ　５Ｂ０４６
	６５４		６５４Ｋ
			６５４Ｍ
	６５６		６５６Ｒ
	６５８		６５８Ｊ

審査請求　未請求　請求項の数５　ＯＬ　（全　８　頁）　最終頁に続く

(21)出願番号　特願2000－361456(P2000－361456)

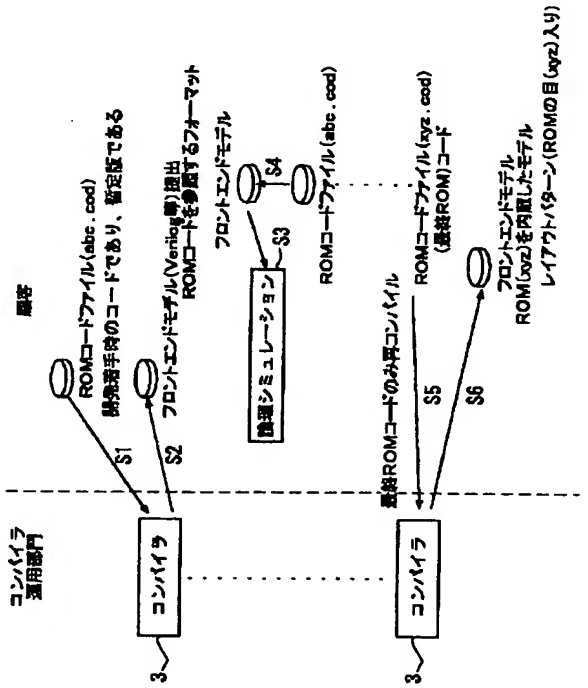
(22)出願日　平成12年11月28日(2000.11.28)

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　藍澤　慎哉
東京都小平市上水本町五丁目20番１号　株式会社日立製作所半導体グループ内
(72)発明者　島田　茂
東京都小平市上水本町五丁目20番１号　株式会社日立製作所半導体グループ内
(74)代理人　100085811
弁理士　大日方　富雄

Ｆターム(参考)　5B046 AA08 BA04 JA05 KA02 KA03 KA06

(54)【発明の名称】　半導体集積回路の設計方法

(57)【要約】
【課題】　論理設計時にＲＯＭモジュールの格納データを修正した場合に、容易に修正後の論理シミュレーションへ移行することが可能であり、格納データのバージョン管理も容易な半導体集積回路の設計方法を提供することにある。
【解決手段】　ＲＯＭモジュールをハードウェア記述言語を用いて論理レベルまで表したデータファイルを生成する際に、ＲＯＭモジュールの記憶セルに記憶されるビットデータの情報をモジュールのデータファイルとは別のファイルに格納し、このビットデータが記録された副データファイルを参照する形式でＲＯＭモジュールのデータファイルを生成する。更に、ＲＯＭモジュールのデータ値を決定づけるコンタクトホールが設けられる層間酸化膜に、格納データが記録されたデータファイルのファイル名が視認可能な態様で刻み込まれるように、ＲＯＭモジュールのマスクパターンのレイアウト設計を行う。



1

【特許請求の範囲】

【請求項１】　ＲＯＭモジュールをハードウェア記述言語を用いて論理レベルまで表したデータファイルを生成する際に、ＲＯＭモジュールの全部又は一部の記憶セルに記憶されるビットデータの情報をモジュールのデータファイルとは別のファイルに格納し、このビットデータが記録された副データファイルを参照する形式でＲＯＭモジュールを論理レベルまで表したデータファイルを生成することを特徴とする半導体集積回路の設計方法。

【請求項２】　半導体集積回路を製造するメーカー側と提供を受ける顧客側との共同により半導体集積回路の設計を行う半導体集積回路の設計方法であって、
メーカーから顧客側に上記ＲＯＭモジュールを論理レベルまで表したデータファイルを提供し、
顧客側で上記ＲＯＭモジュールを表したデータファイルに基づき論理シミュレーションを行うとともに、ＲＯＭモジュールに記憶するビットデータを修正して再度論理シミュレーションする場合に、顧客側で上記ビットデータが記録された副データファイルを変更して、上記ＲＯＭモジュールを表すデータファイルに基づきビットデータ修正後の論理シミュレーションを行うことを特徴とする請求項１記載の半導体集積回路の設計方法。

【請求項３】　顧客側でＲＯＭモジュールに記憶するビットデータが確定した段階で該ビットデータをメーカー側に提示し、
メーカーはこの確定したビットデータに基づき、各記憶セルに記憶されるビットデータの情報を含めた形式でＲＯＭモジュールを論理レベルまで表したデータファイルを生成し、このデータファイルを顧客側に提供することを特徴とする請求項２記載の半導体集積回路の設計方法。

【請求項４】　上記ＲＯＭモジュールは設計時にワード長およびビット長を変更可能なモジュールであることを特徴とする請求項１～３の何れかに記載の半導体集積回路の設計方法。

【請求項５】　ＲＯＭモジュールのマスクパターンのレイアウト設計をする際に、ＲＯＭモジュールの各記憶ビットのデータ値を決定づけるコンタクトホールが設けられる層間絶縁膜のモジュール配置部の近傍に、ＲＯＭモジュールに記憶されるビットデータが記録されたデータファイルを識別する識別情報が視認可能な態様で刻み込まれるように、マスクパターンのレイアウト設計を行うことを特徴とする半導体集積回路の設計方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は、ＲＯＭ（Read Only Memory）モジュールを含んだ半導体集積回路の設計方法に適用して有用な技術に関し、例えばＡＳＩＣ（Application Specific IC）などの半導体集積回路の設計方法に利用して特に有用な技術に関する。

2

【０００２】

【従来の技術】ＬＳＩ設計を行う分野では、ＬＳＩの製造後に手直しを行うと膨大なコストを要するため、設計段階でなるべくバグをなくしておきたいと云う要求が強い。そのため、設計段階でのバグを発見する手段として、設計された回路をコンピュータ上で模擬的に構成し動作させ、この模擬的な回路が所要の機能を果たすか否かを検証するシミュレーション手法が普及している。

【０００３】半導体集積回路のハードウェアの設計には、ＩＣ全体のアーキテクチャやアルゴリズムの設計、機能設計、論理設計、電子回路設計、マスクパターンのレイアウト設計など種々のレベルがあるが、現在、これらの各レベルごとにシミュレーションを行ったり、また、これら複数のレベルを混合させた状態でシミュレーションを行って動作検証を行うことが可能になっている。

【０００４】メーカーと顧客側とで共同してＬＳＩの設計・製造を行うＡＳＩＣの分野においては、顧客側とメーカー側の役割分担が明確に定められており、例えば、アーキテクチャやアルゴリズムの設計からゲートレベルの論理設計および論理シミュレーションまでを顧客側で行い、電子回路設計やマスクパターンのレイアウト設計以降をメーカー側で行うといった設計パターンがある。

【０００５】また、ＡＳＩＣ設計では、ＲＯＭモジュールなどメーカー側から提供される機能回路（マクロセルとも云う）は、顧客にその機能設計データが示されない限り顧客側で論理合成することが出来ないため、一般に、論理シミュレーションへ移行する前に、次のような手順を必要としていた。

【０００６】すなわち、ＡＳＩＣにＲＯＭモジュールを組み込む場合、図５に示すように、顧客側からメーカー（コンパイラ運用部門）へＲＯＭモジュールの仕様（例えばビット長やワード長）や格納データが記録されたＲＯＭコードファイル（ファイル名「ａｂｃ．ｃｏｄ」）を提示し、その後、これらの仕様やその格納データに基づきメーカー側のコンパイラ３でＲＯＭモジュールの論理合成を行い、この論理合成後のモデル（以下、フロントモデルと称する）を顧客側に提供するといった手順を必要としていた。そして、この手順の後に、顧客側で上記フロントモデルを用いて論理シミュレーションを行い、その動作結果から論理検証を行うことが可能となる。

【０００７】

【発明が解決しようとする課題】一般に、ＣＰＵ（Central Processing Unit）やＲＯＭモジュールを内蔵しＣＰＵがＲＯＭモジュールに格納されている命令コードを実行しながら動作するようなＬＳＩでは、論理設計時にＲＯＭモジュールの格納データの修正が頻繁に行われる。

【０００８】しかしながら、従来のＡＳＩＣの設計方法

では、図５に示すように、論理設計時にＲＯＭモジュールの格納データを修正して、修正後の論理シミュレーションを行う場合には、顧客側からメーカー側に修正後の格納データが記録されたＲＯＭコードファイル（「ｏｐｑ．ｃｏｄ」）を再度提示し、メーカー側で論理合成して修正後のフロントモデルを顧客側に提供し直してもらわなければ修正後の論理シミュレーションが行えなかった。この修正後のＲＯＭコードファイルの提示や再度の論理合成といった顧客とメーカー側とのやり取りは面倒で時間がかかるものであり、ＲＯＭモジュールの格納データの修正を繰り返し行えば、その分、上記ＲＯＭコードファイルの提示や論理合成を繰り返し行なわなければならない。

【０００９】さらに、このようにＲＯＭモジュールの格納データを頻繁に修正した場合、最終版として確定された格納データがメーカー側に正しく示されていないと、メーカー側において最終版でない格納データでマスクパターンのレイアウト設計が行われてしまうといった恐れがある。また、マスクパターンに反映されているＲＯＭモジュールの格納データが最終版のものか確認するには、例えば各記憶セルのデータ値を決定するコンタクトホールが設けられる層間絶縁膜のパターンを視認し、このコンタクトホールのレイアウトと最終版の格納データとを逐一比較して手作業で確認するしかなかった。

【００１０】この発明の目的は、論理設計時にＲＯＭモジュールの格納データを修正する場合でも、煩雑な手順を踏まずに容易に修正後の論理シミュレーションへ移行することが可能な半導体集積回路の設計方法を提供することにある。

【００１１】この発明の他の目的は、ＲＯＭモジュールのマスクパターンに反映されている格納データのバージョン確認を容易に行うことのできる半導体集積回路の設計方法を提供することにある。

【００１２】この発明の前記ならびにそのほかの目的と新規な特徴については、本明細書の記述および添附図面から明らかになるであろう。

【００１３】

【課題を解決するための手段】本願において開示される発明のうち代表的なものの概要を説明すれば、下記のとおりである。

【００１４】すなわち、ＲＯＭモジュールをハードウェア記述言語を用いて論理レベルまで表したデータファイルを生成する際に、ＲＯＭモジュールの全部又は一部の記憶セルに記憶されるビットデータの情報をモジュールのデータファイルとは別のファイルに格納し、このビットデータが記録された副データファイルを参照する形式でＲＯＭモジュールを論理レベルまで表したデータファイルを生成する半導体集積回路の設計方法である。

【００１５】このような手段によれば、論理設計時にＲＯＭモジュールに記憶させるビットデータの修正を行って再度論理シミュレーションを行なう場合でも、修正ごとにＲＯＭモジュールの論理合成を行なう必要がなくなり、ＲＯＭモジュールに格納されるビットデータを記録したデータファイルを修正するのみで、修正後の論理シミュレーションを行うことが出来る。従って、論理設計・論理シミュレーションのＴＡＴ（Turn Around Time）を短縮することが出来る。

【００１６】望ましくは、半導体集積回路を製造するメーカー側と提供を受ける顧客側との共同により半導体集積回路の設計を行う半導体集積回路の設計方法であって、メーカーから顧客側に上記ＲＯＭモジュールを論理レベルまで表したデータファイルを提供し、顧客側で上記ＲＯＭモジュールを表したデータファイルに基づき論理シミュレーションを行うとともに、ＲＯＭモジュールに記憶するビットデータを修正して再度論理シミュレーションする場合に、顧客側で上記ビットデータが記録された副データファイルを変更して、上記ＲＯＭモジュールを表すデータファイルに基づきビットデータ修正後の論理シミュレーションを行うようにすると良い。

【００１７】さらに望ましくは、顧客側でＲＯＭモジュールに記憶するビットデータが確定した段階で該ビットデータをメーカー側に提示し、メーカーはこの確定したビットデータに基づき、記憶セルに記憶されるビットデータの情報を含めた形式でＲＯＭモジュールを論理レベルまで表したデータファイルを生成して顧客側に提供すると良い。

【００１８】このような手順によれば、最終的には、格納データの情報まで１つのデータファイルに包含されたＲＯＭモジュールの論理合成モデルが顧客側に示されるので、論理設計以降の過程で、ＲＯＭモジュールの格納データのバージョンを取り違えてしまうといった間違いを無くすことが出来る。

【００１９】具体的には、上記ＲＯＭモジュールは設計時にワード長およびビット長を変更可能なモジュールである。

【００２０】また、ＲＯＭモジュールのマスクパターンのレイアウト設計をする際に、ＲＯＭモジュールの各記憶ビットのデータ値を決定づけるコンタクトホールが設けられる層間絶縁膜のモジュール配置部の近傍に、ＲＯＭモジュールに記憶されるビットデータが記録されたデータファイルを識別する識別情報が視認可能な態様で刻み込まれるように、マスクパターンのレイアウト設計を行うと良い。

【００２１】このような手段によれば、論理設計時にＲＯＭモジュールに格納するデータの修正を頻繁に行ない、マスクパターンのレイアウト設計後にマスクパターンに最終版が正しく反映されているかを確認したい場合でも、マスクパターンにあるビットデータが記録されたデータファイルの識別情報を視認することで、ＲＯＭモジュールに格納されるビットデータが最終版として確定

されたものか否か、容易に確認することが出来る。

【００２２】また、この識別情報の配置を配線等のない領域に設定することで、半導体集積回路の製造後でも、上記識別情報を顕微鏡等を用いて視認し、ＲＯＭモジュールに記憶されているビットデータのデータファイルを確認することが出来る。

【００２３】

【発明の実施の形態】以下、本発明の好適な実施例を図面に基づいて説明する。

【００２４】図１は、本発明を適用して好適な半導体集積回路の設計方法のうち論理設計時におけるＲＯＭモジュールに関する設計データ提供の流れを示す説明図である。同図においては、上から下に向って処理が進められるようになっている。

【００２５】この実施例の半導体集積回路の設計方法は、顧客とメーカーとが協同して設計を行なう例えばＡＳＩＣ（Application Specific ＩＣ）の設計過程で適用されるものであり、具体的には、ＲＯＭモジュールを内蔵するＡＳＩＣにおいて、ＲＯＭモジュールを含めたＡＳＩＣ全体の論理設計および論理シミュレーションを行なう過程に適用されるものである。この実施例では、ＲＯＭモジュールとして、設計時にビット長とワード長を所望の長さに選択できるコンパイルドタイプのマスクＲＯＭモジュールを使用している。

【００２６】図１に示すように、この実施例の設計方法では、先ず、ＡＳＩＣの開発初期の段階において、顧客側からＡＳＩＣに内蔵されるＲＯＭモジュールの基本仕様（ビット長とワード長など）と、ＲＯＭモジュールに格納する格納データ（ＲＯＭコード）が記録されたＲＯＭコードファイルの暫定版（ファイル名「ａｂｄ．ｃｏｄ」）とがメーカー側に提示される（ステップＳ１）。

【００２７】メーカー側に上記データが提示されると、メーカーのコンパイラ運用部門において、ＨＤＬ等で記述され、データベースに予め登録されている当該仕様を有するＲＯＭモジュールの基礎データを読み出して、顧客側からのデータに基づきコンパイラ３を用いてＲＯＭモジュールの論理合成が行われる。以下、論理合成されたＲＯＭモジュールのことをフロントエンドモデルと呼ぶ。そして、このフロントエンドモデルが顧客側に提供される（ステップＳ２）。

【００２８】この実施例では、上記コンパイラ３は、ネットリスト、ＲＴＬ（register transfer level）、或いは論理式で入力された回路を、面積、遅延などの設計制約を満たすよう最適化したり、指定されたライブラリを使って論理合成を行なう論理合成ツールである。コンパイラ３からは、例えば、Ｖｅｒｉｌｏｇ（登録商標）ＨＤＬ（hardware design language）などのハードウェア記述言語を用いて回路を論理レベルまで記述したデータファイルが生成され、顧客側では、このデータファイルを用いて回路の論理シミュレーションを行なうことが可能となる。

【００２９】この実施例においては、初期段階で顧客側に提供されるフロントエンドモデルは、ＲＯＭモジュールに格納される格納データ（ＲＯＭコード）が記録された副データファイルを外付けにして、上記格納データの情報が除かれたＲＯＭモジュールのフロントエンドモデルからこの副データファイルを参照する形式とされる。以下、このモデルのことを参照型フロントエンドモデルと呼ぶ。

【００３０】なお、この実施例では、コンパイラ３による論理合成により、ＲＯＭコードファイルの内容が格納される副データファイルと、ＲＯＭコードファイルの内容が除かれたフロントエンドモデルとの両方が生成される形態になっているが、コンパイラ３からは上記参照型のフロントエンドモデルだけ生成し、副データファイルは所定のフォーマットに従って顧客側で作成するようにしても良い。また、このような形態の場合には、顧客側からメーカー側への暫定的なＲＯＭコードファイルの提示を省くことも出来る。

【００３１】図２には、上記参照型フロントエンドモデルの具体的な記述の一例を示す。

【００３２】同図中、範囲ａの内容は内部変数の時間軸の刻みやバージョン名など、モジュール記載の始まりを参照的に示すものである。ｂ１行の内容はモジュール名と該モジュールにおいて使用される入出力変数、ｂ２行の内容はＲＯＭモジュールのビット長やワード長などのパラメータ、ｂ３行の内容はＲＯＭモジュールの入力信号をそれぞれ示している。また、範囲ｃの内容はモジュール内部で使用する信号名と外部入力される信号名との対応を、範囲ｄの内容はＲＯＭモジュールの動作記述を示している。

【００３３】また、範囲ｅには、指定されたデータファイルのデータを参照するコマンドコードや、参照先のデータファイル名ｅ１、参照したデータが代入される配列変数名“ＭＥＭ”やその始まりの配列番号“０”などが示されている。ここで参照されるデータは、ＲＯＭモジュールに格納される格納データ（ＲＯＭコード）であり、該格納データを欄ｅ１に示されるデータファイルに記録しておくことで、該格納データがモジュール内の配列ＭＥＭ（９’ｂ０００００００００）～ＭＥＭ（９’ｂ１１１１１１１１１）に代入されるようになっている。なお、「９‘ｂ」とは９ビットの値が次に続くことを示している。

【００３４】また、範囲ｆは、ＲＯＭモジュールの記述の終りを示している。

【００３５】参照型フロントエンドモデルが提供されたら、顧客側でＡＳＩＣの論理シミュレーションを行ない（ステップＳ３）、その動作結果に基づき論理検証を行って所望の機能が得られない箇所について論理設計の修正や、ＲＯＭモジュールに格納するファームウェアの修

正などを行う。

【００３６】そして、ＲＯＭモジュールの格納データを修正した場合には、顧客側でこの修正を図２の欄ｅ１に示されるデータファイルに反映させる（ステップＳ４）。それにより、コンパイラ３によるＲＯＭモジュールの再度の論理合成を行なうことなく、この修正を論理合成後のフロントエンドモデルに反映させることが出来る。そして、修正が反映されたフロントエンドモデルを用いて、修正後の論理シミュレーションを行なう（ステップＳ３）。

【００３７】上記のような論理シミュレーション、論理検証、並びに、ファームウェアの修正を繰り返して、最終的な格納データが確定したら、この格納データが記録された最終版のＲＯＭコードファイル（ファイル名「ｘｙｚ．ｃｏｄ」）を顧客側からメーカー側に提示する（ステップＳ５）。

【００３８】最終版のＲＯＭコードファイルが提示されたら、メーカーのコンパイラ運用部門において、初期段階で提示されたＲＯＭモジュールの仕様および最終版のＲＯＭコードファイルに基づき、コンパイラ３を用いてＲＯＭモジュールの論理合成を行ない、このフロントエンドモデルを顧客側に提供する（ステップＳ６）。ここで提供されるフロントエンドモデルは、ＲＯＭモジュールの格納データ（ＲＯＭコード）も１つのデータファイルに包含した形式のものである。以下、このモデルのことを内蔵型フロントエンドモデルと呼ぶ。

【００３９】図３には、上記内蔵型フロントエンドモデルの具体的な記述の一例を示す。

【００４０】この内蔵型フロントエンドモデルの形式は、従来、一般的に用いられているものである。図３において、範囲ａ～ｄ，ｆの内容は図２の参照型フロントエンドモデルのものと同一のものである。

【００４１】範囲ｇでは、ＲＯＭモジュールに格納される格納データが配列変数“ＭＥＭ（９’ｂ０００００００００）”～“ＭＥＭ（９’ｂ１１１１１１１１１）”に代入されるようになっている。すなわち、この格納データが最終版のＲＯＭコードファイルのデータであり、最終版のＲＯＭコードファイルがフロントエンドモデルに内蔵された形式となっている。

【００４２】また、メーカー側のコンパイラ運用部門では、内蔵型のフロントエンドモデルの作成とともに、ＲＯＭモジュールのマスクパターンのレイアウト設計も行なう。

【００４３】図４には、半導体チップのレイアウトパターンの概要図を示す。同図（ａ）は半導体チップ全体の中のＲＯＭモジュールの配置を、（ｂ）はＲＯＭモジュールの部分を示している。

【００４４】ＲＯＭモジュール１０は、ＲＯＭの記憶セルがマトリックス状に配置された記憶セルアレイ１１や、Ｙデコーダやセンスアンプなどの周辺回路等を備えたものである。また、特に制限されるものでないが、この実施の形態のＲＯＭモジュール１０は、ビット線とワード線の各交点の必要な位置にＭＯＳトランジスタを配置して記憶セルを構成し、配線工程においてビット線とＭＯＳトランジスタのドレイン若しくはソースを接続又は非接続とすることで、各記憶セルに記憶されるデータ値が“１”又は“０”に設定されるタイプのマスクＲＯＭである。つまり、ＭＯＳトランジスタとビット線とを接続する層間絶縁膜のコンタクトホールｈ（ｖｉａホールとも呼ぶ）の有無により各記憶セルのデータ値が決定されるものである。

【００４５】さらに、この実施の形態では、図４（ｂ）に示すように、ＲＯＭモジュールのレイアウト設計を行なうときに、コンタクトホールｈが形成される層間絶縁膜において他の配線接続に影響を及ぼさない箇所、例えば、ＲＯＭモジュールの周縁を縁取るように配される電源配線１２と重なる範囲に、このレイアウト設計に組み込まれたＲＯＭコードファイルを識別するための情報としてそのファイル名「ｘｙｚ」がコンタクトホールｈによりドット表示されるように、コンタクトホールｈのレイアウト設計を行う。

【００４６】そして、このようなレイアウト設計を行ったら、コンタクトホールｈのレイアウトパターンが示されたパターンモデルを、上記フロントエンドモデルとともに顧客側に提供する（図１のステップＳ６）。そして、これら内蔵型フロントエンドモデルやパターンモデルに基づき顧客側で最終的な確認作業が行われる。

【００４７】以上のように、この実施例の半導体集積回路の設計方法によれば、論理設計時にＲＯＭモジュールの格納データの修正やその論理シミュレーション並びに論理検証を繰り返し行なう場合でも、格納データの修正ごとにＲＯＭモジュールの論理合成をメーカー側に依頼して行なうという面倒で煩雑な工程が省けるので、メーカー側および顧客側の工数が削減され、論理設計におけるＴＡＴの短縮、延いてはＡＳＩＣ開発期間の短縮を図ることが出来る。

【００４８】また、最終的には、格納データの情報まで１つのデータファイルに包含されたＲＯＭモジュールの論理合成モデルが顧客側に提供されるので、論理設計以降の過程で、ＲＯＭモジュールの格納データのバージョンを取り違えてしまうといった間違いを無くすことが出来る。

【００４９】また、ＲＯＭモジュールのマスクパターンのレイアウト設計をする際に、ＲＯＭモジュールの各記憶ビットのデータ値を決定づけるコンタクトホールｈが設けられる層間絶縁膜に、ＲＯＭモジュールの格納データが記録されたデータファイルのファイル名が視認可能な態様で刻み込まれるように、マスクパターンのレイアウト設計が行われるので、マスクパターンのレイアウト設計後にマスクパターンにＲＯＭコードファイルの最終

版が正しく反映されているかを確認したい場合に、パターンモデルに表示されたＲＯＭコードファイルのファイル名を視認するだけで容易に確認することが出来る。

【００５０】また、このファイル名を表すコンタクトホールｈの配置を配線等のない領域に設定することで、半導体集積回路の製造後でも、上記ファイル名を顕微鏡等を用いて視認し、ＲＯＭモジュールの格納されているＲＯＭコードファイルのバージョン等を確認することが出来る。

【００５１】以上本発明者によってなされた発明を実施例に基づき具体的に説明したが、本発明は上記実施例に限定されるものではなく、その要旨を逸脱しない範囲で種々変更可能であることはいうまでもない。

【００５２】例えば、半導体集積回路に組み込むＲＯＭモジュールとして、コンパイルドタイプのＲＯＭモジュールを例示したが、ビット長やワード長が固定のＲＯＭモジュールであっても同様の効果が得られる。また、最終的にＲＯＭモジュールの格納データも一体化された内蔵型のフロントエンドモデルを顧客側に提供するようにしたが、この内蔵型フロントエンドモデルの論理合成や顧客への提供を省くことも出来る。

【００５３】以上の説明では主として本発明者によってなされた発明をその背景となった利用分野であるＡＳＩＣの設計方法について説明したがこの発明はそれに限定されるものでなく、ＲＯＭモジュールを内蔵する種々の半導体集積回路の設計方法に広く利用することができる。

【００５４】

【発明の効果】本願において開示される発明のうち代表的なものによって得られる効果を簡単に説明すれば下記のとおりである。

【００５５】すなわち、本発明に従うと、論理設計時にＲＯＭモジュールの格納データの修正やその論理シミュレーション並びに論理検証を繰り返し行なう場合でも、格納データの修正ごとにＲＯＭモジュールの論理合成をメーカー側に依頼して行なうという面倒で煩雑な工程が省けるので、メーカー側および顧客側の工数が削減され、論理設計におけるＴＡＴの短縮、延いてはＡＳＩＣ

開発期間の短縮を図ることが出来るという効果がある。

【００５６】また、ＲＯＭモジュールのマスクパターンのレイアウト設計をする際に、ＲＯＭモジュールの各記憶ビットのデータ値を決定づけるコンタクトホールが設けられる層間絶縁膜に、ＲＯＭモジュールの格納データが記録されたデータファイルのファイル名が視認可能な態様で刻み込まれるように、マスクパターンのレイアウト設計が行われるので、マスクパターンのレイアウト設計後にマスクパターンにＲＯＭコードファイルの最終版が正しく反映されているかを確認したい場合に、パターンモデルに表示されたＲＯＭコードファイルのファイル名を視認するだけで容易に確認することが出来るという効果がある。

【図面の簡単な説明】

【図１】本発明を適用して好適な半導体集積回路の設計方法のうち論理設計時におけるＲＯＭモジュールに関する設計データ提供の流れを示す説明図である。

【図２】設計初期の段階でメーカーから顧客側に提供されるデータ参照型のフロントエンドモデルの具体的な記述例を示す参考図である。

【図３】ＲＯＭモジュールの格納データが確定した段階でメーカーから顧客側に提供されるデータ内蔵型のフロントエンドモデルの具体的な記述例を示す参考図である。

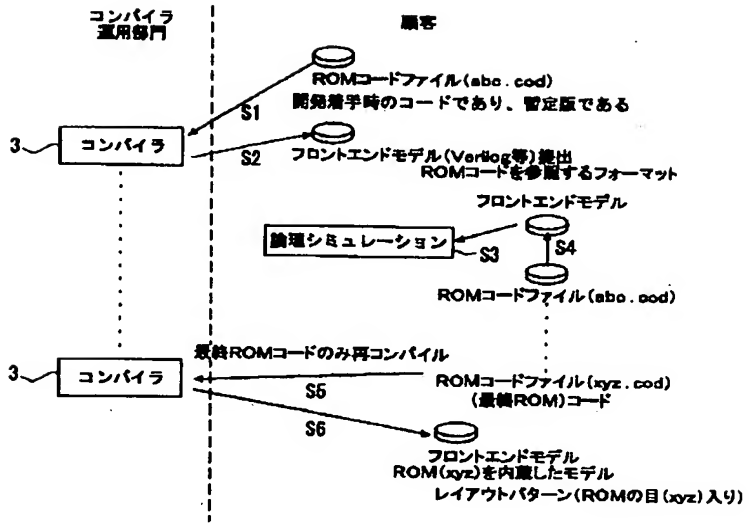
【図４】半導体チップのレイアウトパターンの概要を示すもので、（ａ）は半導体チップ全体の中のＲＯＭモジュールの配置、（ｂ）はＲＯＭモジュールの部分を拡大した図である。

【図５】従来の半導体集積回路の設計方法で、論理設計時におけるＲＯＭモジュールに関するデータ提供の流れを示す説明図である。

【符号の説明】

１０　ＲＯＭモジュール
１２　電源配線
ｈ　コンタクトホール
ｅ　ＲＯＭコードファイルを外部から参照させるコードの記述範囲

【図１】



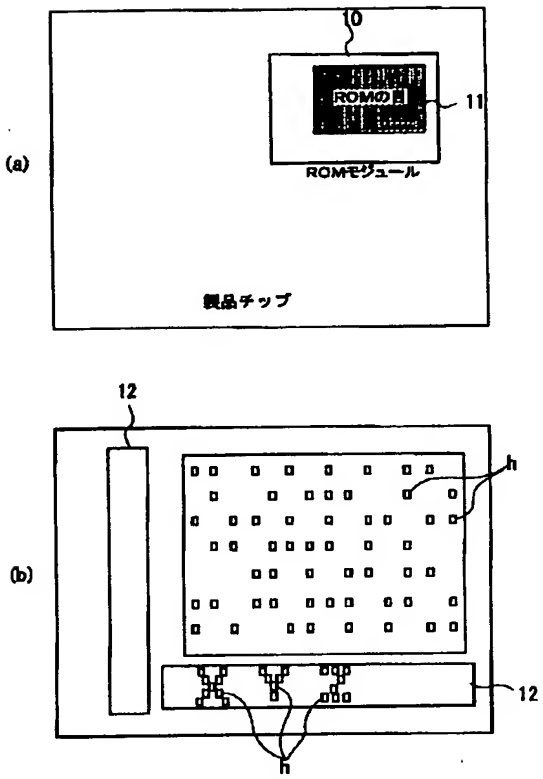
【図２】

```
`timescale 1ns / 10ps `delay_mode_path                    } a
// 76C CROM verilog model version 0.6 2000.04.11
`suppress_faults `enable_portfaults
`celldefine

module rm051216  (a,q,ck,bs,resb);                        b1
parameter numAddr =9,  bits  =16, word_depth=512;         b2
input ck,bs,resb;                                         b3
buf(CK,ck);
buf(BS,bs);                                               } c
buf(RESB,resb);

                                                          } d
initial begin
$readmemb("rm051216.cod",MEM,0);                          } e ← ファイル名の指定
end
                                                          e1
endmodule

`endcelldefine                                            } f
`disable_portfalts
`nosuppress_faults
```

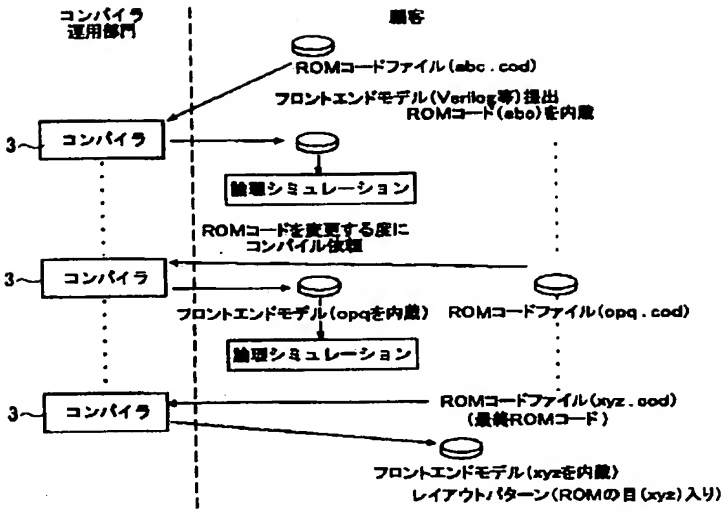
【図３】

```
`timescale 1ns / 10ps `delay_mode_path                    } a
// 76C CROM verilog model version 0.6 2000.04.11
`suppress_faults `enable_portfaults
`celldefine

module rm051216  (a,q,ck,bs,resb);                        } b
parameter numAddr =9,  bits  =16, word_depth=512;
input ck,bs,resb;
buf(CK,ck);
buf(BS,bs);                                               } c
buf(RESB,resb);

                                                          } d
initial begin
MEM[9'b000000000]=16'b1111111111111111;                   } g
MEM[9'b000000001]=16'b1111111111111111;
MEM[9'b000000010]=16'b1111111111111111;
MEM[9'b111111111]=16'b0110011001100110;
end

endmodule

`endcelldefine                                            } f
`disable_portfalts
`nosuppress_faults
```

【図４】



【図５】



フロントページの続き

(51) Int. Cl.[7]	識別記号	FI	テーマコード(参考)
G06F 17/50	658	G06F 17/50	658M